06

Actualizando la branch

Transcripción

[00:00] Hola todos y todas. Bienvenidos y bienvenidas una vez más a este curso de Git. En el último video vimos cómo unir el trabajo de dos branches separadas, de dos líneas de desarrollo, pero yo no quiero generar un commit extra en master. Puedo generar confusión y distracción ya que en el log van a aparecer muchos commits de merge y va a ensuciarlo.

[00:22] Entonces, lo que quiero hacer es actualizar los commits de master con los commits de título. Entonces quiero que los commits que están en titulo, acá, estén antes de mis commits que voy a hacer en master, por ejemplo el de acá. Entonces vamos a hacer ese ejemplo de nuevo. Vamos a dar un clear acá, la consola, aceptamos, y vamos a comenzar de nuevo.

[00:51] Vamos a hacer un git checkout -n titulo. Vamos a generar dos commits, git commit, y ahora vamos a volver a master con git checkout master y vamos a generar un commit, git commit. Perfecto. Entonces lo que ahora quiero es llevar ese commit de master justo después del último commit de título. O sea, ese commit de corrección que hice, de bug, que quede en la parte de adelante de la branch de título. Esa corrección que hice en master que quede delante de título.

[01:37] Entonces estoy en mi master, quiero vaciar mi branch de master en título, o sea quiero hacer un rebase. Lo que voy a hacer es un git rebase título, enter y vean lo que ocurre. ¿Vieron? Lo que hizo fue traer todo lo que son los commits de título y llevarlos antes de nuestro último commit de master. Entonces con esto lo que hice fue generar una única línea, sin esas confusior

de commits de merge extra que nos está creando cada vez que hacemos un git merge

[02:18] Bueno, ahora vamos a la práctica. Volvemos a nuestra, no esa es Ana. Volvemos a Bruno. Entonces voy a hacer un git log, damos enter y vemos que tengo ese commit de merge. ¿Será que puedo ver todos estos commits que habíamos hecho, será que los puedo ver de una forma más interesante dentro de la terminal? Bueno, sí, se puede.

[02:46] Lo que podemos hacer es utilizar el comando git log espacio --graph, damos enter y vean cómo es lo que nos está presentando acá. Nos está diciendo que a partir de este commit comenzamos a corregir las cosas del título, estamos en una nueva branch, una nueva rama. Creamos cosas y finalmente hicimos un merge.

[03:13] Entonces como vemos en la misma, no necesitamos de herramientas externas para poder visualizar nuestras branches. Solamente con tener la terminal es posible hacer eso. Entonces ahora voy a hacer una modificación en mi branch título. Vamos a hacer un git checkout titulo. Enter. Vamos a hacer la modificación dentro de nuestro archivo en Bruno. Acá por ejemplo voy a cambiar curso por la letra C mayúscula. Voy a poner "Ctrl + S" para guardar.

[03:50] Volvemos a nuestra terminal y vamos a agregar ese archivo. Damos un git status para estar seguros, perfecto. Git add index.html. Vamos a hacer un git commit -m "Cursos con la letra mayúscula". Acá me faltó una C. Bien. Damos enter, perfecto, hice el nuevo commit. Ahora quiero ir para mi branch master, entonces hacemos git checkout master.

[04:30] Y ahora lo que quiero hacer es traer los commits de títulos a mi branch master, entonces estando parado en master, voy a hacer un git rebase titulo. Enter. Bien, se ejecutó. Ahora vamos a ver log, si hago un git log, miren lo que ocurre.

[04:49] Tenemos nuestro último commit hecho en master, lo hemos hecho en el video anterior y tenemos nuestro último commit, el que acabamos de hacer hace unos segundo Cursos con la letra mayúscula, fijense que está debajo, o sea que está antes del último commit hecho en master. Entonces esto es exactamente lo que vimos en la herramienta.

[05:13] O sea el rebase actualiza mi branch pero mantiene el trabajo de ella como último trabajo hecho, para que no genere ese tipo de confusión de crear un nuevo merge y que una todo en ese nuevo merge. Con esto tengo las correcciones hechas. Ahora tengo mi título actualizado, los bugs corregidos de la lista y ahora puedo ir a la terminal como Bruno y hacer un push a nuestro repositorio servidor local.

[05.42] Entonces, estando en master vamos a hacer un git push servidorlocal master. Bien, se enviaron los datos a nuestro servidor local. Ahora voy a loguearme como Ana. Vamos acá a Ana y entonces Ana va a hacer un git checkout master, va a hacer un checkout para master y vamos a hacer un pull de lo que tiene un servidor.

[06:23] Vamos a hacer un git pull servidorlocal master. Enter. Con esto tenemos todas las modificaciones que había hecho Bruno. ¿Pero recuerdan que ella estaba trabajando en la lista? Entonces vamos a ir a esa branch, vamos a hacer un git checkout 'lista', enter, y ella necesita actualizar los datos. Abro el proyecto de Ana, bien, acá en el archivo de Ana.

[06:55] Entonces voy al archivo de Ana y sabemos que ese curso de Docker está mal, entonces el nombre verdadero es "Curso de Docker: creando containers sin dolor de cabeza". Vamos a hacer un "Ctrl + S", tengo una actualización acá en mi título. Estoy en el proyecto de Ana y voy a hacer un commit. Entonces git add index.html, vamos a hacer un git commit -m "Actualizando nombre del curso de Docker". Enter.

[07:43] Acá vamos a hacer un "Ctrl+C". En caso que se confundan siempre pueden hacer ahí un "Ctrl + C", parar esa línea que estaban escribiendo y pueden de nuevo realizar ese commit. Hicimos el commit, ahora vamos a hacer un git log -p, enter, vemos que modificamos esa línea, ese curso. Y ahora voy a hacer un checkout hacia master, git checkout master. Y fíjense que tengo una modificación hecha por Bruno y otra modificación hecha por Ana en la misma línea.

[08:31] ¿Qué va a ocurrir si intento juntar el trabajo de los dos? Bueno, vamos a ver lo que ocurre y cómo resolverlo en el próximo video.